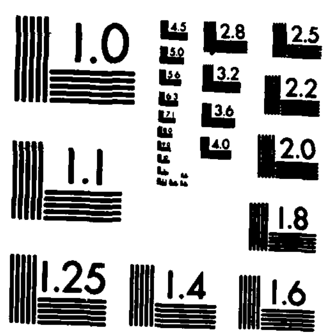
AD-A147 397 USER'S GUIDE FOR QPSOL (VERSION 32): A FORTRAN PACKAGE 1/1
FOR QUADRATIC PROG..(U) STANFORD UNIV CA SYSTEMS
OPTIMIZATION LAB P E GILL ET AL. SEP 84 SOL-84-6
UNCLASSIFIED ARO-21592.1-MA N00014-75-C-0267 F/G 12/1 NL

END


1.0
4.5
5.0
5.6
6.3
7.1
8.0
2.8
3.2
3.6
4.0
2.5
2.2
2.0
1.1
1.8
1.25
1.4
1.6

21592.1-MA

Systems
Optimization
Laboratory

AD-A147 397

User's Guide for QPSOL (Version 3.2)†:
A Fortran Package for Quadratic Programming

by

Philip E. Gill, Walter Murray,
Michael A. Saunders and Margaret H. Wright

TECHNICAL REPORT SOL 84-6
September 1984
(Update of SOL 83-7)

Department of Operations Research
Stanford University
Stanford, CA 94305

SYSTEMS OPTIMIZATION LABORATORY
DEPARTMENT OF OPERATIONS RESEARCH
STANFORD UNIVERSITY
STANFORD, CALIFORNIA 94305

User's Guide for QPSOL (Version 3.2)†: A Fortran Package for Quadratic Programming

by

Philip E. Gill, Walter Murray,
Michael A. Saunders and Margaret H. Wright

TECHNICAL REPORT SOL 84-6

September 1984

(Update of SOL 83-7)

†SOL/QPSOL is available from the Office of Technology Licensing, 105 Encina Hall, Stanford University, Stanford, California, 94305.

Research and reproduction of this report were partially supported by the National Science Foundation Grants MCS-7926009 and ECS-8312142; Department of Energy Contract DE-AM03-76SF00326, PA# DE-AT03-76ER72018; Office of Naval Research Contract N00014-75-C-0267; and Army Research Office Contract DAAG29-84-K-0156.

User's Guide for QPSOL (Version 3.2)†: a Fortran Package for Quadratic Programming

Philip E. Gill, Walter Murray,
Michael A. Saunders and Margaret H. Wright
Systems Optimization Laboratory
Department of Operations Research
Stanford University
Stanford, California 94305

September 1984

ABSTRACT

This report forms the user's guide for Version 3.2 of QPSOL, a set of Fortran subroutines designed to locate the minimum value of a quadratic function subject to linear constraints and simple upper and lower bounds. If the quadratic function is convex, a global minimum is found; otherwise, a local minimum is found. The method used is most efficient when many constraints or bounds are active at the solution. QPSOL treats the Hessian and general constraints as dense matrices, and hence is not intended for large sparse problems.

This document replaces the previous user's guide of July 1983.

† QPSOL is available from the Office of Technology Licensing, 105 Encina Hall, Stanford University, Stanford, California, 94305.

The material contained in this report is based upon research supported by the U.S. Department of Energy Contract DE-AC03-76SF00326, PA No. DE-AT03-76ER72018; National Science Foundation Grants MCS-7926009 and ECS-8012974; the U.S. Army Research Office Contract DAAG29-79-C-0110; and the Office of Naval Research Grant N00014-75-C-0267.

QPSOL User's Guide

TABLE OF CONTENTS

1. PURPOSE

QPSOL is a collection of Fortran subroutines designed to solve the *quadratic programming (QP) problem* — the minimization of a quadratic function subject to a set of linear constraints on the variables. The problem is assumed to be stated in the following form:

$$\text{QP} \qquad \underset{x\in\mathbb{R}^n}{\text{minimize}} \quad c^Tx + \frac{1}{2}x^THx$$
$$\text{subject to} \quad \ell \le \begin{Bmatrix} x \\ \mathcal{A}x \end{Bmatrix} \le u,$$

where c is a constant n-vector and H is a constant $n \times n$ symmetric matrix; note that H is the Hessian matrix (matrix of second partial derivatives) of the quadratic objective function. The matrix $\mathcal{A}$ is $m \times n$, where m may be zero; $\mathcal{A}$ is treated as a dense matrix.

The constraints involving $\mathcal{A}$ will be called the *general* constraints. Note that *upper and lower bounds are specified for all the variables and for all the general constraints.* The form of QP allows full generality in specifying other types of constraints. In particular, an *equality* constraint is specified by setting $\ell_i = u_i$. If certain bounds are not present, the associated elements of ℓ or u can be set to special values that will be treated as $-\infty$ or $+\infty$.

The user must supply an initial estimate of the solution to QP, and a subroutine that computes the product Hx for any given vector x. Some typical examples of this subroutine are included with the QPSOL package. There is no restriction on H apart from symmetry. If H is positive definite or positive semi-definite, QPSOL will obtain a global minimum; otherwise, the solution obtained will be a *local* minimum (which may or may not be a global minimum). If H is defined as the zero matrix, QPSOL will solve the resulting linear programming (LP) problem; however, this can be accomplished more efficiently by setting a logical variable in the call of subroutine **QPSOL** (see the parameter LP in Section 4), or by using the LPSOL package.

QPSOL allows the user to provide the indices of the constraints that are believed to be satisfied exactly at the solution. This facility, known as a *warm start*, can lead to significant savings in computational effort when solving a sequence of related problems. For example, the NPSOL package of Gill *et al.* (1984b) uses this feature in a sequential quadratic programming method for nonlinearly constrained optimization.

The quantity of output is controlled by the user (see the parameter **MSGLVL** discussed in Section 4). The QPSOL package contains approximately 6000 lines of ANSI (1966) Standard Fortran, of which 44% are comments.

2. DESCRIPTION

The method used to solve QP is an *active-set null-space method*, and is described in detail in Gill *et al.* (1984c); a closely related method is given in Gill and Murray (1978). The main features of the method are presented here. Where possible, explicit reference is made to the names of variables that are parameters of subroutine QPSOL or are mentioned in the printed output.

The method has two distinct phases. In the first (the *LP phase*), an iterative procedure is carried out to determine a *feasible point*. In this context, *feasibility* is defined by a user-provided array FEATOL; the j-th constraint is considered satisfied if its violation does not exceed FEATOL(j) (see the discussion of FEATOL in Section 4.) The second phase (the *QP phase*) generates a sequence of feasible iterates in order to minimize the quadratic objective function. In both phases, a subset of the constraints – called the *working set* -- is used to define the search direction at each iteration; typically, the working set includes constraints that are satisfied "exactly" (to within the corresponding tolerances in the FEATOL array).

We now briefly describe a typical iteration in the QP phase. Let x_k denote the estimate of the solution at the k-th iteration; the next iterate is defined by

$$x_{k+1} = x_k + \alpha_k p_k,$$

where p_k is an n-dimensional *search direction* and α_k is a scalar step length. Assume that the working set contains t_k linearly independent constraints, and let C_k denote the matrix of coefficients of the bounds and general constraints in the current working set.

Let Z_k denote a matrix whose columns form a basis for the null space of C_k, so that $C_k Z_k = 0$. (Note that Z_k has n_z columns, where $n_z = n - t_k$.) The vector $Z_k^T(c + Hx_k)$ is called the *projected gradient* at x_k. If the projected gradient is zero at x_k (i.e., x_k is a constrained stationary point in the subspace defined by Z_k), Lagrange multipliers λ_k are defined as the solution of the compatible overdetermined system

$$C_k^T \lambda_k = c + Hx_k. \tag{1}$$

The Lagrange multiplier λ corresponding to an inequality constraint in the working set is said to be *optimal* if $\lambda \leq 0$ when the associated constraint is at its *upper bound*, or if $\lambda \geq 0$ when the associated constraint is at its *lower bound*. If a multiplier is non-optimal, the objective function can be reduced by deleting the corresponding constraint (with index KDEL) from the working set.

If the projected gradient at x_k is nonzero, the search direction p_k is defined as

$$p_k = Z_k p_z, \tag{2}$$

where p_z is an n_z-vector. In effect, the constraints in the working set are treated as *equalities*, by constraining p_k to lie within the subspace of vectors orthogonal to the rows of C_k. This definition ensures that $C_k p_k = 0$, and hence the values of the constraints in the working set are not altered by any move along p_k.

The vector p_z is obtained by solving the equations

$$Z_k^T H Z_k p_z = -Z_k^T (c + H x_k). \tag{3}$$

(The matrix $Z_k^T H Z_k$ is called the *projected Hessian matrix*.) If the projected Hessian is positive definite, the vector defined by (2) and (3) is the step to the minimum of the quadratic function in the subspace defined by Z_k.

If the projected Hessian is positive definite and $x_k + p_k$ is feasible, α_k will be taken as unity. In this case, the projected gradient at x_{k+1} will be zero (see the variable NORM ZTG in the output from QPSOL), and Lagrange multipliers can be computed (see (1)). Otherwise, α_k is set to the step to the "nearest" constraint (with index KADD), which is added to the working set at the next iteration.

The matrix Z_k is obtained from the *TQ factorization* of C_k, in which C_k is represented as

$$C_k Q = \begin{pmatrix} 0 & T_k \end{pmatrix}, \tag{4}$$

where T_k is reverse-triangular (see Gill *et al.*, 1984a). It follows from (4) that Z_k may be taken as the first n_z columns of Q. If the projected Hessian is positive definite, (3) is solved using the *Cholesky factorization*

$$Z_k^T H Z_k = R_k^T R_k,$$

where R_k is upper triangular. These factorizations are *updated* as constraints enter or leave the working set. The update procedures are described in detail in Gill *et al.* (1984a).

An important feature of QPSOL is the treatment of indefiniteness in the projected Hessian. If the projected Hessian is positive definite, it may become indefinite only when a constraint is deleted from the working set. In this case, a temporary modification (of magnitude HESS MOD) is added to the last diagonal element of the Cholesky factor. Once a modification has occurred, no further constraints are deleted from the working set until enough constraints have been added so that the projected Hessian is again positive definite. If problem QP has a finite solution, a move along the direction obtained by solving (3) with the modified Cholesky factor must encounter a constraint that is not already in the working set.

In order to resolve indefiniteness in this way, we must ensure that the projected Hessian is positive definite at the first iterate in the QP phase. Given the $n_z \times n_z$ projected Hessian, a step-wise Cholesky factorization is performed with symmetric interchanges (and corresponding rearrangement of the columns of Z), terminating if the next step would cause the matrix to become indefinite. This determines the largest possible positive-definite principal submatrix of the (permuted) projected Hessian. If n_R steps of the Cholesky factorization have been successfully completed, the relevant projected Hessian is an $n_R \times n_R$ positive-definite matrix $\bar{Z}^T H \bar{Z}$, where $\bar{Z}$ comprises the first n_R columns of Z. The quadratic function will subsequently be minimized within subspaces of reduced dimension until the full projected Hessian is positive definite.

Several strategies are used to control ill-conditioning in the working set. One such strategy is associated with the FEATOL array. Allowing the j-th constraint to be violated by as much as FEATOL(j) often provides a choice of constraints that could be added to the working set. When a choice exists, the decision is based on the conditioning of the working set. Negative steps are occasionally permitted, since x_k may violate the constraint to be added.

3. SPECIFICATION

```
SUBROUTINE QPSOL( ITMAX, MSGLVL, N,
                  NCLIN, NCTOTL, NROWA, NROWH, NCOLH,
                  BIGBND, A, BL, BU, CVEC, FEATOL, HESS, QPHESS,
                  COLD, LP, ORTHOG, ISTATE, X,
                  INFORM, ITER, OBJ, CLAMDA,
                  IW, LENIW, W, LENW )

LOGICAL           COLD, LP, ORTHOG
EXTERNAL          QPHESS
INTEGER           ITMAX, MSGLVL, N, NCLIN, NCTOTL,
                  NROWA, NROWH, NCOLH, INFORM, ITER, LENIW, LENW
INTEGER           ISTATE(NCTOTL), IW(LENIW)
REAL              BIGBND, OBJ
REAL              A(NROWA,N), BL(NCTOTL), BU(NCTOTL), CVEC(N),
                  FEATOL(NCTOTL), HESS(NROWH, NCOLH), X(N),
                  CLAMDA(NCTOTL), W(LENW)
```

Note: Here and elsewhere, the specification of a parameter as REAL should be interpreted as *working precision*, which may be DOUBLE PRECISION in some circumstances.

4. INPUT PARAMETERS

ITMAX is an upper bound on the number of iterations to be taken during the LP phase or the QP phase.

MSGLVL indicates the amount of intermediate output desired. The printout is described in Section 9. All output is written to the file number **NOUT** (see subroutine **MCHPAR** in Section 11). For **MSGLVL** ≥ 10, each value of **MSGLVL** includes the output from all lower values. The printout corresponding to each value of **MSGLVL** is defined as follows:

MSGLVL	Definition
0	No output.
1	The final solution only.
5	One brief line of output for each constraint addition or deletion (no printout of the final solution).
≥ 10	The final solution and one brief line of output for each constraint addition or deletion.
≥ 15	At each iteration, X, ISTATE, and the indices of the free variables (i.e., the variables not currently held on a bound).
≥ 20	At each iteration, the Lagrange multiplier estimates and the general constraint values.
≥ 30	At each iteration, the diagonal elements of the matrix T associated with the TQ factorization of the working set, and the diagonal elements of the Cholesky factor R of the projected Hessian.
≥ 80	Debug printout.
99	The arrays CVEC and HESS.

N is the number of variables (i.e., the dimension of **X**). **N** must be positive.

NCLIN is the number of general linear constraints in the problem (**NCLIN** may be zero).

NCTOTL must be set to **N** + **NCLIN**.

NROWA is the declared row dimension of **A** (**NROWA** must be at least 1 and at least **NCLIN**).

NROWH is the declared row dimension of the array **HESS** (**NROWH** must be at least 1).

NCOLH is the declared column dimension of the array **HESS** (**NCOLH** must be at least 1).

BIGBND is a positive real variable whose magnitude denotes an "infinite" component of ℓ and u. Any upper bound greater than or equal to BIGBND will be regarded as plus infinity (and similarly for a lower bound less than or equal to −BIGBND).

A is a real array of declared dimension (NROWA,N). The i-th row of A contains the coefficients of the i-th general constraint, $i = 1$ to NCLIN. If NCLIN is zero, A is not accessed.

BL is a real array of dimension NCTOTL that contains the lower bounds for all the constraints, in the following order (which is also observed for BU, ISTATE, and CLAMDA): the first N elements of BL contain the lower bounds on the variables; if NCLIN > 0, the next NCLIN elements of BL contain the lower bounds for the general linear constraints. In order for the problem specification to be meaningful, it is required that BL(j) $\leq$ BU(j) for all j. To specify a non-existent lower bound (i.e., $\ell_j = -\infty$), the value used must satisfy BL(j) $\leq$ −BIGBND. To specify the j-th constraint as an *equality*, the user must set BL(j) = BU(j) = β, say, where $|\beta| <$ BIGBND.

BU is a real array of dimension NCTOTL that contains the upper bound · all the constraints, in the same order described above under BL. To specify a no .ent upper bound (i.e., $u_j = \infty$), the value used must satisfy BU(j) $\geq$ BIGBND.

CVEC is a real array of dimension N containing the coefficients of the linear term of the objective function (the vector c in problem QP).

FEATOL is a real array of dimension NCTOTL containing positive tolerances that define the maximum permissible violation in each constraint in order for a point to be considered feasible, i.e. constraint j is considered satisfied if its violation does not exceed FEATOL(j). Note that FEATOL(j) is a bound on the *absolute* acceptable violation. For example, if the data defining the constraints are of order unity and are correct to about 6 decimal digits, it would be appropriate to choose FEATOL(j) as 10^{-6} for all relevant j. In general, the elements of FEATOL should be chosen as the *largest* possible acceptable values, since the algorithm of QPSOL becomes less likely to encounter difficulties with ill-conditioning and degeneracy as the components of FEATOL increase. A warning message is printed if any component of FEATOL is less than machine precision; the user must *not* set any component of FEATOL to zero. A detailed discussion of FEATOL is given in Gill *et al.* (1984c).

HESS is a real array of declared dimension (NROWH, NCOLH) that may be used to store the Hessian matrix H of problem QP if desired. The elements of HESS are accessed only by the subroutine QPHESS; thus HESS is not accessed if LP is .TRUE. In some cases, the user need not use HESS to store H explicitly (see the specification of QPHESS below).

QPHESS is the name of a subroutine that defines the Hessian matrix. QPHESS must be declared as EXTERNAL in the routine that calls QPSOL. QPHESS is not accessed if the logical variable LP is .TRUE. (see the description below of LP). The user has considerable flexibility in coding QPHESS because the algorithm of QPSOL requires only the product of H and a vector; the elements of the matrix H need not be defined explicitly. Several examples of QPHESS are provided in order to demonstrate some of the alternatives. The specification of QPHESS is:

```
SUBROUTINE QPHESS( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
INTEGER            N, NROWH, NCOLH, JTHCOL
REAL               HESS(NROWH, NCOLH), X(N), HX(N).
```

The actual parameters N, NROWH, NCOLH and HESS input to QPHESS will always be the same Fortran variables and arrays as those input to QPSOL. They must *not* be altered by QPHESS.

For a given vector x (the array X), the array HX must contain the product Hx on exit from QPHESS.

The input parameter JTHCOL is included to allow flexibility for the user in the special situation when x is the j-th coordinate vector (i.e., the j-th column of the identity matrix). This may be of interest because the product Hx is then the j-th column of H, which can sometimes be computed very efficiently. The user may code QPHESS to take advantage of this case. If JTHCOL $= j$, where $j > 0$, HX should contain column JTHCOL of H, and hence special code may be included in QPHESS to test JTHCOL if desired. However, special code is *not* necessary, since the vector X always contains column JTHCOL of the identity matrix whenever QPHESS is called with JTHCOL > 0.

In some cases, it may be desirable to use a one-dimensional array to transmit data or workspace to QPHESS; HESS should then be declared as REAL HESS(NROWH), and the parameter NCOLH must be 1. (This device is used for the example subroutines QPHES4 and QPHES6 in the QPSOL package, to economize on storage.)

In other situations, it may be desirable to compute Hx without accessing HESS—for example, if H is sparse or has special structure. (This is illustrated in the subroutine QPHES1 in the QPSOL package.) The parameters HESS, NROWH and NCOLH may then refer to any convenient array.

When MSGLVL $= 99$, the (possibly undefined) contents of HESS will be printed, except if NROWH and NCOLH are both 1. Also printed are the results of calling QPHESS with JTHCOL $= 1, 2, \ldots,$ N.

COLD is a logical variable that indicates whether the user wishes to specify the initial working set. In general, COLD should be set to .TRUE. for the first call of QPSOL, and the initial working set will then be selected by QPSOL. However, if a good estimate of the initial working set is available — for example, when QPSOL is called repeatedly to solve related problems — it may be advantageous to set COLD to .FALSE. after the first call. When COLD is .FALSE., the user must define every element of ISTATE (see the description of ISTATE for the meaning of each possible value). QPSOL will override the user's specification of ISTATE if necessary, so that a poor choice of the working set will not cause a fatal error.

LP is a logical variable. If LP is .FALSE., QPSOL will solve the specified quadratic programming problem. If LP is .TRUE., QPSOL will treat H as zero and solve the resulting linear program; in this case, parameters HESS and QPHESS will not be accessed.

ORTHOG is a logical variable that indicates whether orthogonal transformations are to be used in computing and updating the TQ factorization of the working set

$$AQ = (\,0 \quad T\,),$$

where A is a submatrix of $\mathcal{A}$ and T is reverse-triangular. If ORTHOG is .TRUE., the TQ factorization is computed using Householder reflections and plane rotations, and the matrix Q is orthogonal. If ORTHOG is .FALSE., stabilized elementary transformations are used to maintain the factorization, and Q is not orthogonal. A rule of thumb in making the choice is that orthogonal transformations require more work, but provide greater numerical stability. Thus, we recommend setting ORTHOG to .TRUE. if the problem is reasonably small or the active set is ill-conditioned. Otherwise, setting ORTHOG to .FALSE. will often lead to a reduction in solution time with negligible loss of reliability.

5. INPUT/OUTPUT PARAMETERS

ISTATE is an integer array of dimension NCTOTL that indicates the status of every constraint with respect to the working set. The ordering of ISTATE is the same as that described above under BL, i.e., the first N components of ISTATE refer to the upper and lower bounds on the variables, and components N + 1 through N + NCLIN refer to the upper and lower bounds on Ax. The significance of each possible value of ISTATE(j) is as follows:

ISTATE(j)	Meaning
−2	The constraint violates its lower bound by more than FEATOL(j). This value of ISTATE cannot occur after a feasible point has been found.
−1	The constraint violates its upper bound by more than FEATOL(j). This value of ISTATE cannot occur after a feasible point has been found.
0	The constraint is not in the working set. Usually, this means that the constraint lies strictly between its bounds.
1	This inequality constraint is included in the working set at its lower bound. The value of the constraint is within FEATOL(j) of its lower bound.
2	This inequality constraint is included in the working set at its upper bound. The value of the constraint is within FEATOL(j) of its upper bound.
3	The constraint is included in the working set as an equality. This value of ISTATE can occur only when BL(j) = BU(j). The corresponding constraint is within FEATOL(j) of its required value.

If COLD = .TRUE., ISTATE need not be set by the user. However, when COLD is .FALSE., every element of ISTATE must be set to one of the values given above to define a *suggested* initial working set (which will be changed by QPSOL if necessary). The most likely values are:

ISTATE(j)	Meaning
0	The corresponding constraint should *not* be in the initial working set.
1	The constraint should be in the initial working set at its lower bound.
2	The constraint should be in the initial working set at its upper bound.
3	The constraint should be in the initial working set as an equality. This value must not be specified unless BL(j) = BU(j). The values 1, 2 or 3 all have the same effect when BL(j) = BU(j).

Other values of ISTATE are also acceptable. In particular, if QPSOL has been called previously with the same values of N and NCLIN, ISTATE already contains satisfactory values.

When QPSOL exits with INFORM set to 0, 1 or 3, the values in the array ISTATE indicate the status of the constraints in the active set at the solution. Otherwise, ISTATE indicates the composition of the working set at the final iterate.

X is a real array of dimension N that contains the current estimate of the solution. On entry to QPSOL, X must be defined; on exit from QPSOL, X contains the best estimate of the solution.

6. OUTPUT PARAMETERS

INFORM is an integer that indicates the result of **QPSOL**. (When **MSGLVL** > 0, a short description of **INFORM** is printed.) The possible values of **INFORM** are:

INFORM	Definition
0	X is a strong local minimum, i.e., the projected gradient is negligible, the Lagrange multipliers are optimal, and the projected Hessian is positive definite. In some cases, a zero value of INFORM means that X is a global minimum (e.g., when the Hessian matrix is positive definite).
1	X is a weak local minimum (the projected gradient is negligible, the Lagrange multipliers are optimal, but the projected Hessian is only positive semi-definite). This means that the solution is not unique.
2	The solution appears to be unbounded, i.e., the quadratic function is unbounded below in the feasible region. This value of INFORM occurs when a step of infinity would have to be taken in order to continue the algorithm.
3	X appears to be a local minimum, but optimality cannot be verified because some of the Lagrange multipliers are very small in magnitude.
4	The iterates of the QP phase could be cycling, since a total of 50 changes were made to the working set without altering X.
5	The limit of ITMAX iterations was reached in the QP phase before normal termination occurred.
6	The LP phase terminated without finding a feasible point, and hence it is not possible to satisfy all the constraints to within the tolerances specified by the FEATOL array. In this case, the final iterate will reveal values for which there will be a feasible point (e.g., a feasible point will exist if the feasibility tolerance for each violated constraint exceeds its RESIDUAL at the final point). The modified problem (with altered values in FEATOL) may then be solved using a warm start.
7	The iterates may be cycling during the LP phase; see the comments above under INFORM = 4.
8	The limit of ITMAX iterations was reached during the LP phase.
9	An input parameter is invalid.

ITER is an integer that gives the number of iterations performed in either the LP phase or the QP phase, whichever was last entered. (Note that **ITER** is reset to zero after the LP phase.)

OBJ is the value of the quadratic objective function at X if X is feasible (INFORM $\leq$ 5), or the sum of infeasibilities at X otherwise (6 $\leq$ INFORM $\leq$ 8).

CLAMDA is a real array of dimension NCTOTL that contains the Lagrange multiplier for every constraint with respect to the current working set. The ordering of CLAMDA follows the convention given above under BL, i.e., the first N components contain the multipliers for the bound constraints on the variables, and the remaining components contain the multipliers for the general linear constraints. If ISTATE(j) = 0 (i.e., constraint j is not in the working set), CLAMDA(j) is zero. If X is optimal, CLAMDA(j) should be non-negative if ISTATE(j) = 1 and non-positive if ISTATE(j) = 2.

7. WORKSPACE PARAMETERS

IW is an integer array of dimension **LENIW**, which provides integer workspace for **QPSOL**.

LENIW is the dimension of **IW**, and must be at least $N + 2 + \min(N, NCLIN)$.

W is a real array of dimension **LENW**, which provides real workspace for **QPSOL**.

LENW is the dimension of **W**. If **LP** = .FALSE. or $NCLIN \geq N$, **LENW** must be at least $2N^2 + 4N + NROWA + 2NCON$, where $NCON = \max(1, NCLIN)$. If **LP** = .TRUE. and $NCLIN < N$, **LENW** must be at least $2NCON^2 + 4N + NROWA + 2NCON$.

If $MSGLVL > 0$, the amounts of workspace provided and required are printed. As an alternative to computing **LENW** from the formula given above, the user may prefer to obtain an appropriate value from the output of a preliminary run with a positive value of **MSGLVL** and **LENW** set to 1 (**QPSOL** will then terminate with **INFORM** = 9).

8. AUXILIARY SUBPROGRAMS AND LABELLED COMMON

The subroutines associated specifically with the QPSOL package are the following:

ADDCON	ALLOC	BDPERT	BNDALF
CHKDAT	DELCON	FINDP	GETLAM
LPBGST	LPCORE	LPCRSH	LPDUMP
LPGRAD	LPPRT	MOVEX	QPCHKP
QPCOLR	QPCORE	QPCRSH	QPDUMP
QPGRAD	QPPRT	PRTSOL	RSOLVE
TQADD	TSOLVE	ZYPROD.	

QPSOL also uses the basic linear algebra subroutines

AXPY	CONDVC	COPYMX	COPYVC
DOT	DSCALE	ELM	ELMGEN
ETAGEN	QUOTNT	REFGEN	ROT3
ROTGEN	SSCALE	V2NORM	ZEROVC

and the subroutine MCHPAR, which defines machine-dependent constants (see Section 11).

The subroutines in the QPSOL package use the following labelled COMMON areas:

SOLMCH (15 REAL variables; see Section 11)
SOL1CM (3 INTEGER variables)
SOL3CM (4 INTEGER variables)
SOL4CM (10 REAL variables)
SOL5CM (3 REAL variables)
SOL1LP (15 INTEGER variables)
SOL2LP (1 LOGICAL variable.)

9. DESCRIPTION OF THE PRINTED OUTPUT

This section describes the intermediate printout produced by QPSOL. When MSGLVL ≥ 5, a line of output is produced for every change in the working set (thus, several lines may be printed during a single iteration).

To aid interpretation of the printed results, we mention the convention for numbering the constraints: indices 1 through N refer to the bounds on the variables, and indices N + 1 through N + NCLIN refer to the general constraints. When the status of a constraint changes, the index of the constraint is printed, along with the designation "L" (lower bound), "U" (upper bound) or "E" (equality). If the problem is non-convex, the character "V" may appear alongside an index in the "delete" column. This will occur if the initial projected Hessian is not sufficiently positive definite (and therefore the Cholesky factor corresponds only to a *subset* of the columns of Z; see Section 2). The "V" is used to indicate that the Cholesky factor has been expanded to include a new column of Z. The associated index gives the current dimension of the Cholesky factor.

In the LP phase, the printout includes the following:

ITN is the iteration count.

KDEL is the index of the constraint deleted from the working set. If KDEL is zero, no constraint was deleted.

KADD is the index of the constraint added to the working set. If KADD is zero, no constraint was added.

STEP is the step taken along the computed search direction.

NUMINF is the number of violated constraints (infeasibilities).

SUMINF is a weighted sum of the magnitudes of the constraint violations.

LPOBJ is the value of the linear objective function $c^T x$. It is printed only if LP is .TRUE.

During the QP phase, the printout includes the following:

ITN is the iteration count (reset to zero after the LP phase).

KDEL is the index of the constraint deleted from the working set. If KDEL is zero, no constraint was deleted.

KADD is the index of the constraint added to the working set. If KADD is zero, no constraint was added.

STEP is the step α_k taken along the direction of search (if **STEP** is 1.0, the current point is a minimum in the subspace defined by the current working set).

NHESS is the number of calls to subroutine **QPHESS**.

OBJECTIVE is the value of the quadratic objective function.

NCOLZ is the number of columns of Z (see Section 2). In general, it is the dimension of the subspace in which the quadratic is currently being minimized.

NORM GFREE is the Euclidean norm of the gradient of the objective function with respect to the free variables, i.e. variables not currently held at a bound (**NORM GFREE** is not printed if **ORTHOG** is **.FALSE.**). In some cases, the objective function and gradient are updated rather than recomputed. If so, this entry will be "--" to indicate that the gradient with respect to the free variables has not been computed.

NORM QTG is a weighted norm of the gradient of the objective function with respect to the free variables (**NORM QTG** is not printed if **ORTHOG** is **.TRUE.**). In some cases, the objective function and gradient are updated rather than recomputed. If so, this entry will be "--" to indicate that the gradient with respect to the free variables has not been computed.

NORM ZTG is the Euclidean norm of the projected gradient (see Section 2).

HESS MOD is the correction added to the diagonal of the projected Hessian to ensure that a satisfactory Cholesky factorization exists (see Section 2). When the projected Hessian is sufficiently positive definite, **HESS MOD** will be zero.

When **MSGLVL** $= 1$ or **MSGLVL** ≥ 10, the summary printout at the end of execution of **QPSOL** includes a listing of the status of every constraint. Note that default names are assigned to all variables and constraints.

The following describes the printout for each variable.

VARIABLE is the name (**VARBL**) and index j of the variable.

STATE gives the state of the variable (**FR** if neither bound is in the working set, **EQ** if a fixed variable, **LL** if on its lower bound, **UL** if on its upper bound). If **VALUE** lies outside the upper or lower bounds by more than **FEATOL**(j), **STATE** will be "++" or "--" respectively.

VALUE is the value of the variable at the final iteration.

LOWER BOUND is the lower bound specified for the variable. ("NONE" indicates that BL(j) $\leq$ −BIGBND.)

UPPER BOUND is the upper bound specified for the variable. ("NONE" indicates that BU(j) $\geq$ BIGBND.)

LAGR MULTIPLIER is the value of the Lagrange multiplier for the associated bound constraint. This will be zero if STATE is FR. If X is optimal, the multiplier should be non-negative if STATE is LL, and non-positive if STATE is UL.

RESIDUAL is the difference between the variable and the nearer of its bounds BL(j) and BU(j).

The following summary printout is given for each general constraint.

LINEAR CONSTR is the name (LNCON) and index i, $i = 1$ to NCLIN, of the constraint.

STATE is the state of the constraint (FR for a constraint not in the working set, EQ for an equality, LL for an inequality constraint at its lower bound, UL for an inequality constraint at its upper bound). If VALUE lies outside the upper or lower bounds by more than its feasibility tolerance, STATE will be "++" or "--" respectively.

VALUE is the value of the constraint at the final point, i.e., the appropriate component of the vector Ax.

LOWER BOUND is the specified lower bound for the constraint. ("NONE" indicates that BL(N + i) $\leq$ −BIGBND.)

UPPER BOUND is the specified upper bound for the constraint. ("NONE" indicates that BU(N + i) $\geq$ BIGBND.)

LAGR MULTIPLIER is the value of the Lagrange multiplier. This will be zero if STATE is FR. If X is optimal, the multiplier should be non-negative if STATE is LL, and non-positive if STATE is UL.

RESIDUAL is the residual of the constraint with respect to its nearer bound, i.e., the difference between VALUE and the nearer of its two bounds.

10. ERROR RECOVERY

Reason for termination	Recommended Action
Underflow	If the machine parameter indicating an underflow check (WMACH(9)) is zero, floating-point underflow may occur occasionally, but can usually be ignored. To avoid underflow, set WMACH(9) to a positive value; however, this will lead to a noticeable loss of efficiency. If underflow continues to occur for no apparent reason, contact the authors at Stanford University.
Overflow	If the printed output before the overflow error contains a warning about serious ill-conditioning in the working set when adding the j-th constraint, it may be possible to avoid the difficulty by increasing the magnitude of FEATOL(j) and rerunning the program. If the message recurs even after this change, the offending linearly dependent constraint (with index "j") must be removed from the problem. If a warning message did not precede the fatal overflow, contact the authors at Stanford University.
INFORM $= 3$	QPSOL has probably found a solution. However, the presence of very small Lagrange multipliers means that the predicted active set may be incorrect, or that X may be only a constrained stationary point rather than a local minimum. The method in QPSOL is not *guaranteed* to find the correct active set when there are small multipliers. QPSOL attempts to delete constraints with zero multipliers, but this does not necessarily resolve the issue. The determination of the correct active set is a combinatorial problem that may require an extremely large amount of time. The occurrence of small multipliers often (*but not always*) indicates that there are redundant constraints.
INFORM $= 4$	This value will occur if 50 iterations are performed in the QP phase without changing X. The user should check the printed output for a repeated pattern of constraint deletions and additions. If a sequence of constraint changes is being repeated, the iterates are probably cycling. (QPSOL does not contain a method that is *guaranteed* to avoid cycling, which would be combinatorial in nature.) Cycling may occur in two circumstances: at a constrained stationary point where there are some small or zero Lagrange multipliers (see the discussion of INFORM $= 3$); or at a point (usually a vertex) where the constraints that are satisfied

exactly are nearly linearly dependent. In the latter case, the user has the option of identifying the offending dependent constraints and removing them from the problem, or restarting the run with larger values of FEATOL for nearly dependent constraints. If QPSOL terminates with INFORM = 4, but no suspicious pattern of constraint changes can be observed, it may be worthwhile to restart with the final X (with or without the warm start option).

INFORM = 5 — The value of ITMAX may be too small. If the method appears to be making progress (e.g., the objective function is being satisfactorily reduced), increase ITMAX and rerun QPSOL (possibly using the warm start facility to specify the initial working set). If ITMAX is already large, but some of the constraints could be nearly linearly dependent, check the output for a repeated pattern of constraints entering and leaving the working set. (Near-dependencies are often indicated by wide variations in size in the diagonal elements of the T matrix, which will be printed if MSGLVL $\geq$ 30.) In this case, the algorithm could be cycling (see the comments for INFORM = 4.)

INFORM = 6 — The LP phase has terminated without finding a feasible point, which means that no feasible point exists for the given FEATOL array. The user should check that there are no constraint redundancies. If the data for the j-th constraint are accurate only to the absolute precision δ, the user should ensure that the value of FEATOL(j) is *greater* than δ. For example, if all elements of A are of order unity and are accurate only to three decimal places, every component of FEATOL should be at least 10^{-3}.

INFORM= 7 or 8 — These values are the analogue in the LP phase procedure of INFORM values 4 and 5.

11. IMPLEMENTATION INFORMATION

This program has been written in ANSI (1966) Fortran and tested on an IBM 3081 computer using the WATFIV Compiler Version 1 Level 6. All subroutines in QPSOL are PFORT-compatible (Ryder, 1974), except for CHKDAT, GETLAM and PRTSOL, which contain A2 format specifications.

At the beginning of QPSOL, the subprogram MCHPAR is called to assign various machine-dependent parameters. These parameters are stored in the array WMACH(15) in the labelled COMMON block SOLMCH.

The specification of MCHPAR is

```
SUBROUTINE MCHPAR
REAL                WMACH
COMMON    /SOLMCH/  WMACH(15)
```

The first eleven components of the REAL array WMACH must be set in MCHPAR. The components of WMACH are defined as follows.

	Definition
WMACH(1)	is NBASE, the base of floating-point arithmetic.
WMACH(2)	is NDIGIT, the number of NBASE digits of precision.
WMACH(3)	is EPSMCH, the floating-point precision.
WMACH(4)	is RTEPS, the square root of EPSMCH.
WMACH(5)	is FLMIN, the smallest positive floating-point number.
WMACH(6)	is RTMIN, the square root of FLMIN.
WMACH(7)	is FLMAX, the largest positive floating-point number.
WMACH(8)	is RTMAX, the square root of FLMAX.
WMACH(9)	is UNDFLW, which specifies whether or not NPSOL should check for underflow in certain computations. If UNDFLW = 0, no underflow checking will be performed. If UNDFLW is set to a positive number, QPSOL will check for underflow and will replace too-small quantities by zero. *Note that* QPSOL *will run faster if no underflow checking takes place*, i.e. if WMACH(9) = 0.0.
WMACH(10)	is NIN, the file number for the input stream.
WMACH(11)	is NOUT, the file number for the output stream.

The following version of MCHPAR (which is provided by the Systems Optimization Laboratory) contains the parameters associated with double precision on a machine in the IBM 370 series. *The user must substitute a version of* MCHPAR *that is appropriate for the machine to be used.*

```
      SUBROUTINE MCHPAR
C
      DOUBLE PRECISION   WMACH
      COMMON    /SOLMCH/ WMACH(15)
C
C  MCHPAR  MUST DEFINE THE RELEVANT MACHINE PARAMETERS AS FOLLOWS.
C     WMACH(1)  = NBASE  = BASE OF FLOATING-POINT ARITHMETIC.
C     WMACH(2)  = NDIGIT = NO. OF BASE WMACH(1) DIGITS OF PRECISION.
C     WMACH(3)  = EPSMCH = FLOATING-POINT PRECISION.
C     WMACH(4)  = RTEPS  = SQRT(EPSMCH).
C     WMACH(5)  = FLMIN  = SMALLEST POSITIVE FLOATING-POINT NUMBER.
C     WMACH(6)  = RTMIN  = SQRT(FLMIN).
C     WMACH(7)  = FLMAX  = LARGEST POSITIVE FLOATING-POINT NUMBER.
C     WMACH(8)  = RTMAX  = SQRT(FLMAX).
C     WMACH(9)  = UNDFLW = 0.0 IF UNDERFLOW IS NOT FATAL, +VE OTHERWISE.
C     WMACH(10) = NIN    = STANDARD FILE NUMBER OF THE INPUT STREAM.
C     WMACH(11) = NOUT   = STANDARD FILE NUMBER OF THE OUTPUT STREAM.
C
      INTEGER            NBASE, NDIGIT, NIN, NOUT
      DOUBLE PRECISION   DSQRT
C
      NBASE     = 16
      NDIGIT    = 14
      WMACH(1)  = NBASE
      WMACH(2)  = NDIGIT
      WMACH(3)  = WMACH(1)**(1 - NDIGIT)
      WMACH(4)  = DSQRT(WMACH(3))
      WMACH(5)  = WMACH(1)**(-62)
      WMACH(6)  = DSQRT(WMACH(5))
      WMACH(7)  = WMACH(1)**61
      WMACH(8)  = DSQRT(WMACH(7))
      WMACH(9)  = 0.0D+0
      NIN       = 5
      NOUT      = 6
      WMACH(10) = NIN
      WMACH(11) = NOUT
C
C---- IN WATFIV, ALLOW UP TO 100 UNDERFLOWS.
C---- CALL TRAPS ( 0,0,100 )
      RETURN
C
C  END OF MCHPAR
      END
```

The values of **NBASE**, **NDIGIT**, **EPSMCH**, **FLMIN** and **FLMAX** for several machines are given in the following table, for both single and double precision; **RTEPS**, **RTMIN** and **RTMAX** may be computed using Fortran statements. The values **NIN** and **NOUT** depend on the machine installation.

For each precision, we give two values for **EPSMCH**, **FLMIN** and **FLMAX**. The first value is a Fortran decimal approximation of the exact quantity; use of this value in **MCHPAR** should cause no difficulty except in extreme circumstances. The second value is the exact mathematical representation.

Table of machine-dependent parameters

Variable	IBM 360/370 Single	CDC 6000/7000 Single	DEC 10/20 Single	Univac 1100 Single	DEC VAX Single
NBASE	16	2	2	2	2
NDIGIT	6	48	27	27	24
EPSMCH	9.54E-7 16^{-5}	7.11E-15 2^{-47}	7.46E-9 2^{-27}	1.50E-8 2^{-26}	1.20E-7 2^{-23}
FLMIN	1.0E-78 16^{-65}	1.0E-293 2^{-975}	1.0E-38 2^{-129}	1.0E-38 2^{-129}	1.0E-38 2^{-128}
FLMAX	1.0E+75 $16^{63}(1-16^{-6})$	1.0E+322 $2^{1070}(1-2^{-48})$	1.0E+38 $2^{127}(1-2^{-27})$	1.0E+38 $2^{127}(1-2^{-27})$	1.0E+38 $2^{127}(1-2^{-24})$

Variable	IBM 360/370 Double	CDC 6000/7000 Double	DEC 10/20 Double	Univac 1100 Double	DEC VAX Double
NBASE	16	2	2	2	2
NDIGIT	14	96	62	61	56
EPSMCH	2.22D-16 16^{-13}	2.53D-29 2^{-95}	2.17D-19 2^{-62}	8.68D-19 2^{-60}	2.78D-17 2^{-55}
FLMIN	1.0D-78 16^{-65}	1.0D-293 2^{-975}	1.0D-38 2^{-129}	1.0D-308 2^{-1025}	1.0D-38 2^{-128}
FLMAX	1.0D+75 $16^{63}(1-16^{-14})$	1.0D+322 $2^{1070}(1-2^{-96})$	1.0D+38 $2^{127}(1-2^{-62})$	1.0D+307 $2^{1023}(1-2^{-61})$	1.0D+38 $2^{127}(1-2^{-56})$

12. EXAMPLE PROGRAM AND OUTPUT

This section contains a listing and the computed results from a sample main program that calls QPSOL to solve an indefinite quadratic program. The problem has seven variables and seven general constraints.

The vector c is given by

$$c = (-.02,\ -.2,\ -.2,\ -.2,\ -.2,\ .04,\ .04)^T.$$

The Hessian is

$$H = \begin{pmatrix} 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 2 & 2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2 & -2 \\ 0 & 0 & 0 & 0 & 0 & -2 & -2 \end{pmatrix},$$

and is defined by the subroutine QPHES1, which does not store H explicitly.

The general constraint matrix A is

$$A = \begin{pmatrix} 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 & 1.0 \\ .15 & .04 & .02 & .04 & .02 & .01 & .03 \\ .03 & .05 & .08 & .02 & .06 & .01 & 0.0 \\ .02 & .04 & .01 & .02 & .02 & 0.0 & 0.0 \\ .02 & .03 & 0.0 & 0.0 & .01 & 0.0 & 0.0 \\ .70 & .75 & .80 & .75 & .80 & .97 & 0.0 \\ .02 & .06 & .08 & .12 & .02 & .01 & .97 \end{pmatrix}.$$

The lower and upper bound vectors ℓ and u are

$$\begin{aligned} \ell = (&-.01,\ -.1,\ -.01,\ -.04,\ -.1,\ -.01,\ -.01, \\ &-.13,\ -\infty,\ -\infty,\ -\infty,\ -\infty,\ -.099,\ -.003\ )^T, \\ u = (&.01,\ .15,\ .03,\ .02,\ .05,\ +\infty,\ +\infty, \\ &-.13,\ -.0049,\ -.0064,\ -.0037,\ -.0012,\ +\infty,\ .002\ )^T. \end{aligned}$$

The starting point x_0 (which is infeasible) is

$$x_0 = (-.01,\ -.03,\ 0.0,\ -.01,\ -.1,\ .02,\ .01)^T.$$

The computed solution (to five figures) is

$$x^* = (-.01,\ -.069865,\ .018259,\ -.024261,\ -.062006,\ .013805,\ .0040665\ )^T.$$

One bound constraint and four general constraints are active at the solution.

```
C*****BEGIN FILE  QPMAIN FORTRAN D.
C
C  EXAMPLE PROGRAM FOR SUBROUTINE QPSOL.
C  DOUBLE PRECISION VERSION 3.2.  SEPTEMBER 1984.
C  THE VALUE OF THE PARAMETER   FEATOL  IS APPROPRIATE FOR A MACHINE
C  WITH A PRECISION OF 15  DECIMAL DIGITS.
C  ******************************************************************
      INTEGER            I, INFORM, ITER, ITMAX, J, LIWORK, LWORK
      INTEGER            MSGLVL, N, NCLIN, NCOLH, NCOLH1, NCTOTL
      INTEGER            NIN, NOUT, NROWA, NROWH, NROWH1
      INTEGER            ISTATE(14), IWORK(50)
      DOUBLE PRECISION   BIGBND, EPSMCH, OBJ, RTEPS
      DOUBLE PRECISION   ZERO, TWO
      DOUBLE PRECISION   A(7,7), BL(14), BU(14), CLAMDA(14), CVEC(7)
      DOUBLE PRECISION   FEATOL(14), HESS(1,1), HESS1(7,7), X(7)
      DOUBLE PRECISION   WORK(200)
      DOUBLE PRECISION   DSQRT
      LOGICAL            COLD, LP, ORTHOG
      EXTERNAL           QPHES1, QPHES2
      DATA               ZERO  , TWO
     *                  /0.0D+0, 2.0D+0/
C
C  SET THE DECLARED ARRAY DIMENSIONS.
C  NROWA  = THE DECLARED ROW DIMENSION OF  A.
C  NROWH  = THE DECLARED ROW DIMENSION OF  HESS.
C  NCOLH  = THE NUMBER OF COLUMNS IN  HESS.
C           (IF  QPHESS  DEALS WITH THE HESSIAN IMPLICITLY,
C           NROWH  AND  NCOLH  CAN BOTH BE 1.)
C  LIWORK = THE LENGTH OF THE INTEGER WORK ARRAY.
C  LWORK  = THE LENGTH OF THE DOUBLE PRECISION WORK ARRAY.
C
      NROWA  = 7
      NROWH  = 1
      NCOLH  = 1
      LIWORK = 50
      LWORK  = 200
C
C  SET THE APPROXIMATE MACHINE PRECISION.
C
      EPSMCH = 1.0D-15
C
C  ALLOW UP TO 20 ITERATIONS TO FIND A FEASIBLE POINT,
C  AND THE SAME NUMBER TO MINIMIZE THE QUADRATIC FUNCTION.
C
      ITMAX  = 20
C
C  ASK FOR BRIEF OUTPUT EACH ITERATION, AND A FULL PRINT-OUT
C  OF THE FINAL SOLUTION.
C
      MSGLVL = 10
C
C  SET THE PROBLEM DIMENSIONS.
C  N      = THE NUMBER OF VARIABLES.
C  NCLIN  = THE NUMBER OF GENERAL LINEAR CONSTRAINTS (MAY BE 0).
C  NCTOTL = THE TOTAL NUMBER OF VARIABLES AND GENERAL CONSTRAINTS.
C           (THE ARRAYS  ISTATE, BL, BU, CLAMBDA  MUST BE AT LEAST
C           THIS LONG.)
C
```

```
      N      = 7
      NCLIN  = 7
      NCTOTL = N + NCLIN
C
C  BOUNDS GREATER THAN  BIGBND  WILL BE TREATED AS PLUS  INFINITY.
C  BOUNDS LESS THAN   - BIGBND  WILL BE TREATED AS MINUS INFINITY.
C
      BIGBND = 1.0E+10
C
C  ANY BOUND OR LINEAR CONSTRAINT MAY BE VIOLATED BY AS MUCH AS  FEATOL.
C
      RTEPS  = DSQRT( EPSMCH )
      DO 20 J = 1, NCTOTL
         FEATOL(J) = RTEPS
   20 CONTINUE
C
C  A COLD START IS NEEDED FOR THE FIRST CALL TO  QPSOL.
C  WE WANT TO SOLVE A QUADRATIC PROGRAM, NOT AN LP PROBLEM.
C  USE AN ORTHOGONAL FACTORIZATION OF THE MATRIX OF CONSTRAINTS
C  IN THE WORKING SET.
C
      COLD   = .TRUE.
      LP     = .FALSE.
      ORTHOG = .TRUE.
C
C  READ THE DATA ARRAYS.
C  NIN    = THE UNIT NUMBER FOR INPUT.
C  NOUT   = THE UNIT NUMBER FOR PRINTING.
C  CVEC   = THE LINEAR PART OF THE OBJECTIVE FUNCTION.
C  A      = THE GENERAL CONSTRAINT MATRIX.
C  BL     = THE LOWER BOUNDS ON  X  AND  A*X.
C  BU     = THE UPPER BOUNDS ON  X  AND  A*X.
C  X      = THE INITIAL ESTIMATE OF THE SOLUTION.
C
      NIN    = 5
      NOUT   = 6
      READ (NIN, 1000) ( CVEC(J), J=1,N )
      READ (NIN, 1000) (( A(I,J), J=1,N ), I=1,NCLIN )
      READ (NIN, 1000) (    BL(J), J=1,NCTOTL )
      READ (NIN, 1000) (    BU(J), J=1,NCTOTL)
      READ (NIN, 1000) (     X(J), J=1,N)
C
C  PRINT THE DATA.
C
      IF (NOUT .LE. 0) GO TO 50
      WRITE (NOUT, 2000) (CVEC(J), J=1,N)
      WRITE (NOUT, 2100) ((A(I,J), J=1,N), I=1,NCLIN)
      WRITE (NOUT, 2200) (  BL(J), J=1,NCTOTL)
      WRITE (NOUT, 2300) (  BU(J), J=1,NCTOTL)
      WRITE (NOUT, 2400) (   X(J), J=1,N)
C
C
C  SOLVE THE PROBLEM.
C  THE HESSIAN IS DEFINED IMPLICITLY BY SUBROUTINE  QPHES1.
C
   50 CALL QPSOL( ITMAX, MSGLVL, N,
     *            NCLIN, NCTOTL, NROWA, NROWH, NCOLH,
     *            BIGBND, A, BL, BU, CVEC, FEATOL, HESS, QPHES1,
     *            COLD, LP, ORTHOG, ISTATE, X,
     *            INFORM, ITER, OBJ, CLAMDA,
     *            IWORK, LIWORK, WORK, LWORK )
```

```
C
C  TEST FOR AN ERROR CONDITION.
C
      IF (INFORM .GT. 0) GO TO 900
C
C
C  THE FOLLOWING IS FOR ILLUSTRATIVE PURPOSES ONLY.
C  WE DO A WARM START WITH THE FINAL WORKING SET OF THE PREVIOUS RUN.
C  THIS TIME WE STORE THE HESSIAN EXPLICITLY IN  HESS1,
C  AND USE THE CORRESPONDING SUBROUTINE  QPHES2.
C
      WRITE (NOUT, 2500)
      COLD   = .FALSE.
      MSGLVL = 5
      NROWH1 = 7
      NCOLH1 = 7
C
      DO 200 J = 1, N
         DO 100 I = 1, N
            HESS1(I,J) = ZERO
  100    CONTINUE
         IF (J .LE. 5) HESS1(J,J) =   TWO
         IF (J .GT. 5) HESS1(J,J) = - TWO
  200 CONTINUE
C
      HESS1(3,4) =   TWO
      HESS1(4,3) =   TWO
      HESS1(6,7) = - TWO
      HESS1(7,6) = - TWO
C
      CALL QPSOL( ITMAX, MSGLVL, N,
     *            NCLIN, NCTOTL, NROWA, NROWH1, NCOLH1,
     *            BIGBND, A, BL, BU, CVEC, FEATOL, HESS1, QPHES2,
     *            COLD, LP, ORTHOG, ISTATE, X,
     *            INFORM, ITER, OBJ, CLAMDA,
     *            IWORK, LIWORK, WORK, LWORK )
C
      IF (INFORM .GT. 0) GO TO 900
      STOP
C
C  ERROR EXIT.
C
  900 WRITE (NOUT, 3000) INFORM
      STOP
C
 1000 FORMAT(7E10.2)
 2000 FORMAT(/ 14H CVEC.         / (1X, 7F10.2))
 2100 FORMAT(/ 14H ROWS OF  A.   / (1X, 7F10.2))
 2200 FORMAT(/ 14H LOWER BOUNDS. / (1X, 7E10.2))
 2300 FORMAT(/ 14H UPPER BOUNDS. / (1X, 7E10.2))
 2400 FORMAT(/ 12H INITIAL  X.   / (1X, 7F10.2))
 2500 FORMAT(//48H A RUN OF THE SAME EXAMPLE WITH A WARM START....)
 3000 FORMAT(/ 32H QPSOL TERMINATED WITH  INFORM =, I3)
C
C  END OF THE EXAMPLE PROGRAM FOR QPSOL.
      END

      SUBROUTINE QPHES1( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
```

```
      DOUBLE PRECISION   HESS(NROWH,NCOLH), HX(N), X(N)
C
C  ------------------------------------------------------------------
C  QPHES1  COMPUTES THE VECTOR  HX = (HESS)*X  FOR SOME MATRIX  HESS
C  THAT DEFINES THE HESSIAN OF THE REQUIRED QP PROBLEM.
C
C  IN THIS VERSION OF  QPHESS  THE HESSIAN MATRIX IS IMPLICIT.
C  THE ARRAY  HESS  IS NOT ACCESSED.  THERE IS NO SPECIAL CODING
C  FOR THE CASE  JTHCOL .GT. 0.
C  ------------------------------------------------------------------
C
      DOUBLE PRECISION   ONE, TWO
      DATA               ONE/1.0D+0/, TWO/2.0D+0/
C
      HX(1) = TWO*X(1)
      HX(2) = TWO*X(2)
      HX(3) = TWO*(X(3) + X(4))
      HX(4) = HX(3)
      HX(5) = TWO*X(5)
      HX(6) = - TWO*(X(6) + X(7))
      HX(7) = HX(6)
      RETURN
C
C  END OF QPHES1
      END

      SUBROUTINE QPHES2( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
      DOUBLE PRECISION   HESS(NROWH,NCOLH), HX(N), X(N)
C
C  ------------------------------------------------------------------
C  IN THIS VERSION OF QPHESS, THE MATRIX  H  IS STORED IN  HESS  AS
C  A FULL TWO-DIMENSIONAL ARRAY.
C  COPYVC  AND  ZEROVC  ARE UTILITY ROUTINES USED BY  QPSOL.
C  ------------------------------------------------------------------
C
      INTEGER            I, J
      DOUBLE PRECISION   XJ
C
      IF (JTHCOL .EQ. 0) GO TO 100
C
C  SPECIAL CASE -- EXTRACT ONE COLUMN OF  H.
C
      CALL COPYVC( N, HESS(1,JTHCOL), N, 1, HX, N, 1 )
      RETURN
C
C  NORMAL CASE.
C
  100 CALL ZEROVC( N, HX, N, 1 )
      DO 200 J = 1, N
         XJ = X(J)
         DO 150 I = 1, N
            HX(I) = HX(I) + HESS(I,J)*XJ
  150    CONTINUE
  200 CONTINUE
      RETURN
C
C  END OF QPHES2
      END
```

```
      SUBROUTINE QPHES3( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
      DOUBLE PRECISION   HESS(NROWH,NCOLH), HX(N), X(N)
C
C ---------------------------------------------------------------
C IN THIS VERSION OF QPHESS, THE SYMMETRIC PART OF  H  IS STORED IN
C THE LOWER HALF OF THE TWO-DIMENSIONAL ARRAY  HESS,  I.E., IN THE
C ELEMENTS  HESS(I,J),  I .GE. J.
C ---------------------------------------------------------------
C
      INTEGER            I, J, JP1, LROWH, NM1, NUM
      DOUBLE PRECISION   S, XJ
C
      IF (JTHCOL .EQ. 0) GO TO 100
C
C SPECIAL CASE -- EXTRACT ONE COLUMN OF  H.
C
      LROWH  = NROWH*(JTHCOL - 1) + 1
      CALL COPYVC( JTHCOL, HESS(JTHCOL,1), LROWH, NROWH, HX, JTHCOL, 1 )
      NUM    = N - JTHCOL
      JP1    = JTHCOL + 1
      IF (NUM .GT. 0)
     *   CALL COPYVC( NUM, HESS(JP1,JTHCOL), NUM, 1, HX(JP1), NUM, 1 )
      RETURN
C
C NORMAL CASE.
C
  100 DO 200 I = 1, N
         S = 0.0D+0
         DO 150 J = I, N
            S = S + HESS(J,I)*X(J)
  150    CONTINUE
         HX(I) = S
  200 CONTINUE
      IF (N .LE. 1) RETURN
C
      NM1 = N - 1
      DO 400 J = 1, NM1
         XJ  = X(J)
         JP1 = J + 1
         DO 350 I = JP1, N
            HX(I) = HX(I) + HESS(I,J)*XJ
  350    CONTINUE
  400 CONTINUE
      RETURN
C
C  END OF QPHES3
      END

      SUBROUTINE QPHES4( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
      DOUBLE PRECISION   HESS(NROWH), HX(N), X(N)
C
C ---------------------------------------------------------------
C  IN THIS VERSION OF QPHESS, THE SYMMETRIC PART OF  H  IS STORED IN
C  THE ONE-DIMENSIONAL ARRAY  HESS.  NOTE THAT  NROWH  IS USED TO DEFINE
C  THE LENGTH OF  HESS,  AND MUST BE AT LEAST  N*(N + 1)/2.  THE
C  PARAMETER  NCOLH  IS NOT USED HERE, BUT IT MUST BE SET TO  1  FOR
C  THE CALL TO QPSOL.
C ---------------------------------------------------------------
```

```
C
      INTEGER            I, INC, J, JP1, L, NM1, NUM
      DOUBLE PRECISION   S, XJ
C
      IF (JTHCOL .EQ. 0) GO TO 100
C
C  SPECIAL CASE -- EXTRACT ONE COLUMN OF  H.
C
      L      = JTHCOL
      INC    = N
      DO 50 I = 1, JTHCOL
         HX(I) = HESS(L)
         INC   = INC - 1
         L     = L + INC
   50 CONTINUE
C
      L      = L - INC + 1
      NUM    = N - JTHCOL
      JP1    = JTHCOL + 1
      IF (NUM .GT. 0)
     *   CALL COPYVC( NUM, HESS(L), NUM, 1, HX(JP1), NUM, 1 )
      RETURN
C
C  NORMAL CASE.
C
  100 L = 0
      DO 200 I = 1, N
         S = 0.0D+0
         DO 150 J = I, N
            L = L + 1
            S = S + HESS(L)*X(J)
  150    CONTINUE
         HX(I) = S
  200 CONTINUE
      IF (N .LE. 1) RETURN
C
      L   = 0
      NM1 = N - 1
      DO 400 J = 1, NM1
         XJ  = X(J)
         L   = L + 1
         JP1 = J + 1
         DO 350 I = JP1, N
            L     = L + 1
            HX(I) = HX(I) + HESS(L)*XJ
  350    CONTINUE
  400 CONTINUE
      RETURN
C
C  END OF QPHES4
      END

      SUBROUTINE QPHES5( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
      DOUBLE PRECISION   HESS(NROWH,NCOLH), HX(N), X(N)
C
C  ------------------------------------------------------------------
C  IN THIS VERSION OF QPHESS, THE CHOLESKY FACTOR OF  H  IS STORED IN
C  THE LOWER HALF OF THE TWO-DIMENSIONAL ARRAY  HESS.  IN OTHER WORDS,
C  H = L * L(TRANSPOSE),  WHERE  L  IS A LOWER TRIANGULAR MATRIX STORED
```

```
C  IN  HESS(I,J),  I .GE. J.
C  ------------------------------------------------------------------
C
      INTEGER            I, IBACK, J, JMAX, LROWH, NUM
      INTEGER            MIN0
      DOUBLE PRECISION   S
C
      IF (JTHCOL .EQ. 0) GO TO 100
C
C  SPECIAL CASE -- WE NEED  HX = L * (JTH ROW OF L).
C
      NUM    = N - JTHCOL + 1
      CALL ZEROVC( NUM, HX(JTHCOL), NUM, 1 )
      NUM    = JTHCOL
      LROWH  = NROWH*(NUM - 1) + 1
      CALL COPYVC( NUM, HESS(JTHCOL,1), LROWH, NROWH, HX, NUM, 1 )
      GO TO 300
C
C  NORMAL CASE.
C
  100 DO 200 I = 1, N
         S = 0.0D+0
         DO 150 J = I, N
            S = S + HESS(J,I)*X(J)
  150    CONTINUE
         HX(I) = S
  200 CONTINUE
C
      NUM   = N
C
C  COMPUTE  HX = L * HX.
C
  300 IBACK = N
      DO 400 I = 1, N
         S     = 0.0D+0
         JMAX  = MIN0( NUM, IBACK )
         DO 350 J = 1, JMAX
            S     = S + HESS(IBACK,J)*HX(J)
  350    CONTINUE
         HX(IBACK) = S
         IBACK     = IBACK - 1
  400 CONTINUE
      RETURN
C
C  END OF QPHESS
      END

      SUBROUTINE QPHES6( N, NROWH, NCOLH, JTHCOL, HESS, X, HX )
      INTEGER            N, NROWH, NCOLH, JTHCOL
      DOUBLE PRECISION   HESS(NROWH), HX(N), X(N)
C
C  ------------------------------------------------------------------
C  IN THIS VERSION OF QPHESS, THE CHOLESKY FACTOR OF  H  IS STORED IN
C  THE ONE-DIMENSIONAL ARRAY  HESS.  IN OTHER WORDS,
C  H = L * L(TRANSPOSE),  WHERE  L  IS A LOWER TRIANGULAR MATRIX STORED
C  COMPACTLY BY COLUMNS IN  HESS.  NOTE THAT  NROWH  IS USED TO DEFINE
C  THE LENGTH OF  HESS,  AND MUST BE AT LEAST  N*(N + 1)/2.  THE
C  PARAMETER  NCOLH  IS NOT USED HERE, BUT IT SHOULD BE SET TO  1  FOR
C  THE CALL TO QPSOL.
C  ------------------------------------------------------------------
```

```
C
      INTEGER            I, IBACK, INC, J, JMAX, L, NUM
      INTEGER            MIN0
      DOUBLE PRECISION   S
C
      IF (JTHCOL .EQ. 0) GO TO 100
C
C  SPECIAL CASE -- WE NEED  HX = L * (JTH ROW OF L).
C
      NUM    = N - JTHCOL + 1
      CALL ZEROVC( NUM, HX(JTHCOL), NUM, 1 )
      L      = JTHCOL
      INC    = N
      DO 50 I = 1, JTHCOL
         HX(I) = HESS(L)
         INC   = INC - 1
         L     = L + INC
   50 CONTINUE
C
      NUM    = JTHCOL
      GO TO 300
C
C  NORMAL CASE.
C
  100 L = 0
      DO 200 I = 1, N
         S = 0.0D+0
         DO 150 J = I, N
            L = L + 1
            S = S + HESS(L)*X(J)
  150    CONTINUE
         HX(I) = S
  200 CONTINUE
C
      NUM    = N
C
C  COMPUTE  HX = L * HX.
C
  300 IBACK = N
      DO 400 I = 1, N
         S      = 0.0D+0
         L      = IBACK
         INC    = N
         JMAX   = MIN0( NUM, IBACK )
         DO 350 J = 1, JMAX
            S      = S + HESS(L)*HX(J)
            INC    = INC - 1
            L      = L + INC
  350    CONTINUE
         HX(IBACK) = S
         IBACK     = IBACK - 1
  400 CONTINUE
      RETURN
C
C  END OF QPHES6
      END
```

```
CVEC.
     -0.02      -0.20      -0.20      -0.20      -0.20       0.04       0.04

ROWS OF  A.
      1.00       1.00       1.00       1.00       1.00       1.00       1.00
      0.15       0.04       0.02       0.04       0.02       0.01       0.03
      0.03       0.05       0.08       0.02       0.06       0.01       0.00
      0.02       0.04       0.01       0.02       0.02       0.00       0.00
      0.02       0.03       0.00       0.00       0.01       0.00       0.00
      0.70       0.75       0.80       0.75       0.80       0.97       0.00
      0.02       0.06       0.08       0.12       0.02       0.01       0.97

LOWER BOUNDS.
 -0.10D-01 -0.10D 00 -0.10D-01 -0.40D-01 -0.10D 00 -0.10D-01 -0.10D-01
 -0.13D 00 -0.10D 13 -0.10D 13 -0.10D 13 -0.10D 13 -0.99D-01 -0.30D-02

UPPER BOUNDS.
  0.10D-01  0.15D 00  0.30D-01  0.20D-01  0.50D-01  0.10D 13  0.10D 13
 -0.13D 00 -0.49D-02 -0.64D-02 -0.37D-02 -0.12D-02  0.10D 13  0.20D-02

INITIAL  X.
     -0.01      -0.03       0.00      -0.01      -0.10       0.02       0.01

WORKSPACE PROVIDED IS      IW(     50),  W(    200).
TO SOLVE PROBLEM WE NEED   IW(     14),  W(    161).


 ITN JDEL  JADD        STEP    COND T NUMINF        SUMINF
   0    0     0  0.00D-01  1.83D 02       3  1.038000D-01
   1   9U   13L  4.12D-02  1.56D 02       1  3.000000D-02
   2  12U    4L  4.24D-02  5.30D 01       0  0.000000D-01

EXIT LP PHASE.   INFORM =  0   ITER =   2


 ITN JDEL  JADD        STEP NHESS   OBJECTIVE NCOLZ NORM GFREE  NORM ZTG   COND T COND ZHZ  HESS MOD
   0    0     0  0.00D-01     1  4.5800D-02     0    2.41D-01  0.00D-01  5.3D 01  1.0D 00  0.00D-01
   0   5L     0  0.00D-01     2  4.5800D-02     1    4.67D-01  2.16D-01  6.0D 01  1.0D 00  0.00D-01
   1    0   14L  1.33D-01     3  4.1616D-02     0    4.44D-01  0.00D-01  6.0D 01  1.0D 00  0.00D-01
   1  11U     0  0.00D-01     4  4.1616D-02     1    4.44D-01  9.46D-02  1.3D 01  1.0D 00  0.00D-01
   2    0     0  1.00D 00     5  3.9362D-02     1    4.33D-01  1.39D-17  1.3D 01  1.0D 00  0.00D-01
   2   3L     0  0.00D-01     6  3.9362D-02     2    5.26D-01  9.20D-02  1.5D 01  1.3D 00  0.00D-01
   3    0   10U  4.15D-01     7  3.7589D-02     1    5.18D-01  1.19D-02  5.7D 01  1.0D 00  0.00D-01
   4    0     0  1.00D 00     8  3.7554D-02     1    5.18D-01  3.47D-18  5.7D 01  1.0D 00  0.00D-01
   4   4L     0  0.00D-01     9  3.7554D-02     2    5.77D-01  5.01D-02  5.3D 01  1.2D 00  0.00D-01
   5    0     0  1.00D 00    10  3.7032D-02     2    5.57D-01  8.59D-18  5.3D 01  1.2D 00  0.00D-01

EXIT QP PHASE.   INFORM =  0   ITER =   5


VARIABLE        STATE      VALUE        LOWER BOUND      UPPER BOUND  LAGR MULTIPLIER     RESIDUAL

VARBL  1          LL  -0.10000000D-01  -0.10000000D-01  0.10000000D-01   0.4700306         0.0000
VARBL  2          FR  -0.69864665D-01  -0.1000000       0.1500000        0.0000000         0.3014D-01
VARBL  3          FR   0.18259150D-01  -0.10000000D-01  0.30000000D-01   0.0000000         0.1174D-01
VARBL  4          FR  -0.24260810D-01  -0.40000000D-01  0.20000000D-01   0.0000000         0.1574D-01
VARBL  5          FR  -0.62005640D-01  -0.1000000       0.50000000D-01   0.0000000         0.3799D-01
VARBL  6          FR   0.13805440D-01  -0.10000000D-01      NONE         0.0000000         0.2381D-01
VARBL  7          FR   0.40664960D-02  -0.10000000D-01      NONE         0.0000000         0.1407D-01
```

```
LINEAR CONSTR   STATE      VALUE          LOWER BOUND      UPPER BOUND    LAGR MULTIPLIER      RESIDUAL

LNCON  1          EQ  -0.1300000       -0.1300000       -0.1300000        -1.908183         0.4163D-16
LNCON  2          FR  -0.5879898D-02       NONE         -0.4900000D-02   0.0000000          0.9799D-03
LNCON  3          UL  -0.6400000D-02       NONE         -0.6400000D-02  -0.3143604          0.8674D-18
LNCON  4          FR  -0.4537323D-02       NONE         -0.3700000D-02   0.0000000          0.8373D-03
LNCON  5          FR  -0.2915996D-02       NONE         -0.1200000D-02   0.0000000          0.1716D-02
LNCON  6          LL  -0.9920000D-01  -0.9920000D-01        NONE          1.954501          0.5551D-16
LNCON  7          LL  -0.3000000D-02  -0.3000000D-02   0.2000000D-02    1.971586          0.2711D-18

EXIT QPSOL - OPTIMAL QP SOLUTION.

FINAL QP OBJECTIVE VALUE =    0.37031165D-01


A RUN OF THE SAME EXAMPLE WITH A WARM START....

WORKSPACE PROVIDED IS      IW(     50),  W(    200).
TO SOLVE PROBLEM WE NEED   IW(     14),  W(    161).

EXIT LP PHASE.   INFORM =  0   ITER =    0


 ITN JDEL  JADD       STEP NHESS  OBJECTIVE NCOLZ NORM GFREE  NORM ZTG   COND T COND ZHZ  HESS MOD
   0    0     0  0.00D-01     3  3.7032D-02     2    5.57D-01  8.65D-16  3.5D 01  1.3D 00  0.00D-01

EXIT QP PHASE.   INFORM =  0   ITER =    0

EXIT QPSOL - OPTIMAL QP SOLUTION.

FINAL QP OBJECTIVE VALUE =    0.37031165D-01
```

REFERENCES

Gill, P. E., Murray, W., Saunders, M. A., and Wright, M. H. (1984a). Procedures for optimization problems with a mixture of bounds and general linear constraints, ACM *Transactions on Mathematical Software* **10**.

Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. (1984b). User's guide for NPSOL (Version 2.1): a Fortran package for nonlinear programming, Report SOL 84-5, Department of Operations Research, Stanford University, California.

Gill, P. E., Murray, W., Saunders, M. A. and Wright, M. H. (1984c). The design and implementation of a quadratic programming algorithm, to appear, Department of Operations Research, Stanford University, California.

Gill, P. E., Murray, W. and Wright, M. H. (1981). *Practical Optimization*, Academic Press, London and New York.

Ryder, B. G. (1974). The PFORT verifier, *Software-Practice and Experience* **4**, pp. 359–377.

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: ARO 21592.1-MA
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle): User's Guide for QPSOL (Version 3.2): Package for Quadratic Programming
5. TYPE OF REPORT & PERIOD COVERED: Technical Report
6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s): Philip E. Gill, Walter Murray, Michael A. Saunders and Margaret H. Wright
8. CONTRACT OR GRANT NUMBER(s): N00014-75-C-0267; DAAG29-84-K-0156
9. PERFORMING ORGANIZATION NAME AND ADDRESS: Department of Operations Research - SOL, Stanford University, Stanford, CA 94305
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: NR-047-143
11. CONTROLLING OFFICE NAME AND ADDRESS: Office of Naval Research - Dept. of the Navy, 800 N. Quincy Street, Arlington, VA 22217

    U.S. Army Research Office, P.O. Box 12211, Research Triangle Park, NC 27709
12. REPORT DATE: September 1984
13. NUMBER OF PAGES: 36 pp.
15. SECURITY CLASS. (of this report): UNCLASSIFIED
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report)

    This document has been approved for public release and sale; its distribution is unlimited.
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)
18. SUPPLEMENTARY NOTES
19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

    optimization; mathematical software; quadratic programming; linear programming

20. ABSTRACT (Continue on reverse side if necessary and identify by block number) This report forms the user's guide for Version 3.2 of QPSOL, a set of Fortran subroutines designed to locate the minimum value of an arbitrary quadratic function subject to linear constraints and simple upper and lower bounds. If the quadratic function is convex, a global minimum is found; otherwise, a local minimum is found. The method used is most efficient when many constraints or bounds are active at the solution. QPSOL treats the Hessian and general constraints as dense matrices, and hence is not intended for large sparse problems. This document replaces the previous user's guide of July 1983.

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

END

FILMED

12-84

DTIC